V4L2 Video Decoder/Encoder Development Journey

April 2024

Nas Chung
Senior Software Engineer
Chips&Media

Contents

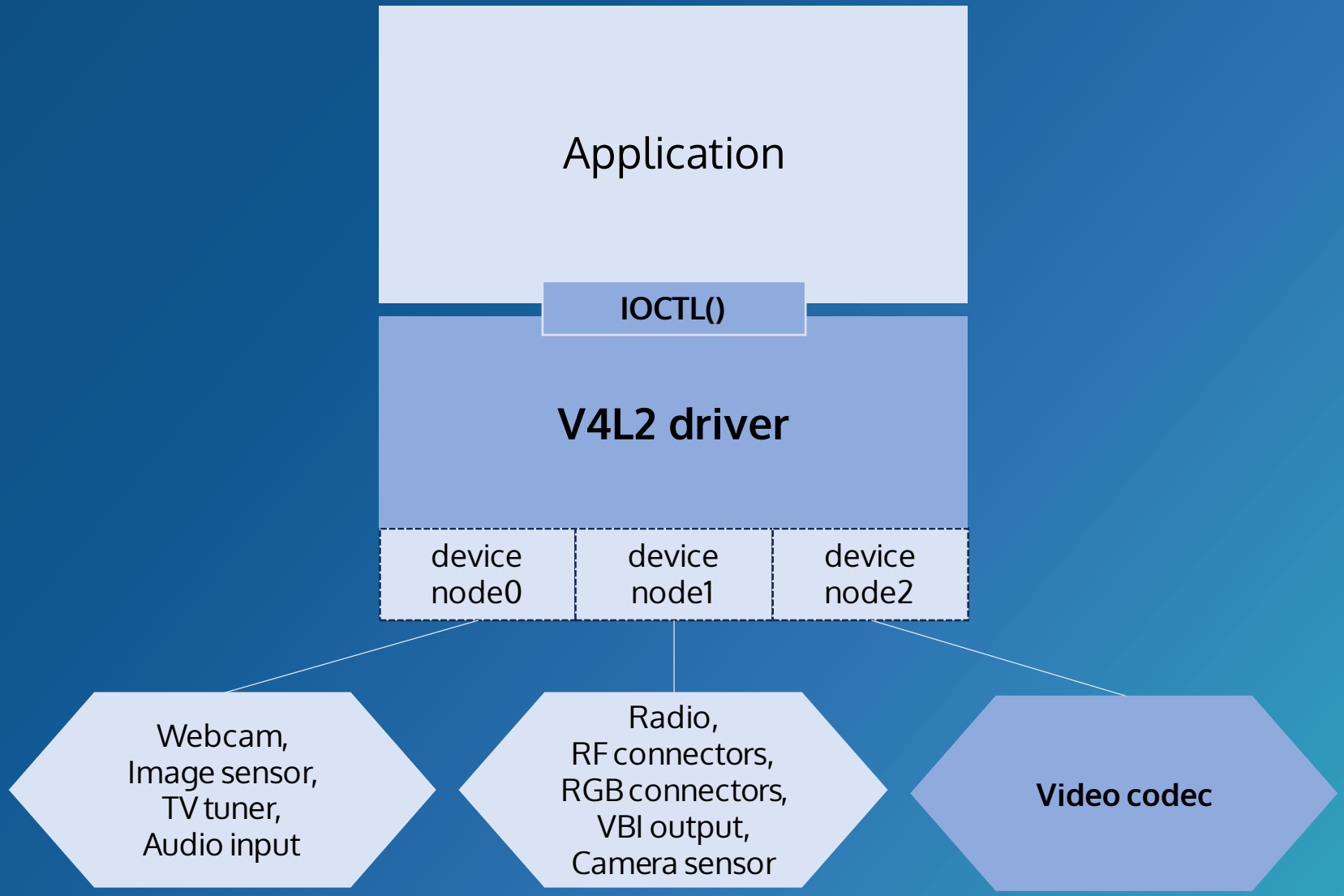
- Introduction
- V4L2 specification
- Initial development on FPGA board
- Verification on TI target board
- Upstreaming process

Introduction

Nas Chung

- Senior software engineer in Chips&Media
- C&M API for video codec HW
  - Coda9, Wave5/6, Etc
- Experience with Multimedia framework
  - VA-API media driver
  - V4L2 driver
  - Wave5 driver is merged in linux-kernel mainline
    - work with Collabora engineers

Study V4L2 specification



Study V4L2 specification

V4L2 IOCTL flow

VIDIOC_QUERYCAP()	Obtain information for video device capabilities
VIDIOC_S_FMT()	Set buffer format information
VIDIOC_REQBUFS()	Request to allocate buffers
VIDIOC_QUERYBUF()	Obtain information for the allocated buffers
VIDIOC_QBUF()	Enqueue the buffer
VIDIOC_STREAMON()	Start streaming
VIDIOC_DQBUF()	Dequeue the buffer
VIDIOC_STREAMOFF()	Finish streaming

For CAPTURE_TYPE
- Device => Host

For OUTPUT_TYPE
- Host => Device

Study V4L2 specification

VideoBuf2 interface

- Video buffer memory allocation and management
  - dma-sg : physically scattered
  - vmalloc : vmalloc() buffer
  - dma-contig : physically contiguous
- Don't need to copy buffer to user space
- Video buffer queue, dequeue and state management

Study V4L2 specification

Memory-to-memory(mem2mem, m2m) interface

- Related with video codec, scalers, format converters
- Support both OUTPUT and CAPTURE stream I/O
- Support multiple contexts
- If both buffers are available, handle m2m_job

Study V4L2 specification

Event interface

- A generic way to pass events to user space
- Event is stored in each event queue
- If (event num > queue size) remove oldest event

Study V4L2 specification

Control interface

- Provide user-specific IOCTL command to fit their device
- Extended control interface is available
  - For encoding MPEG/VPX/HEVC

Study V4L2 specification

Video codec

- Decoder
  - Compressed data (Bitstream) => Raw data (YUV, Pixel data, Image)
- Encoder
  - Raw data => Compressed data (Bitstream)
- Need input buffer and output buffer

Study V4L2 specification

V4L2 IOCTL flow for stateful decoder

VIDIOC_S_FMT(OUT)

VIDIOC_REQBUFS(OUT)

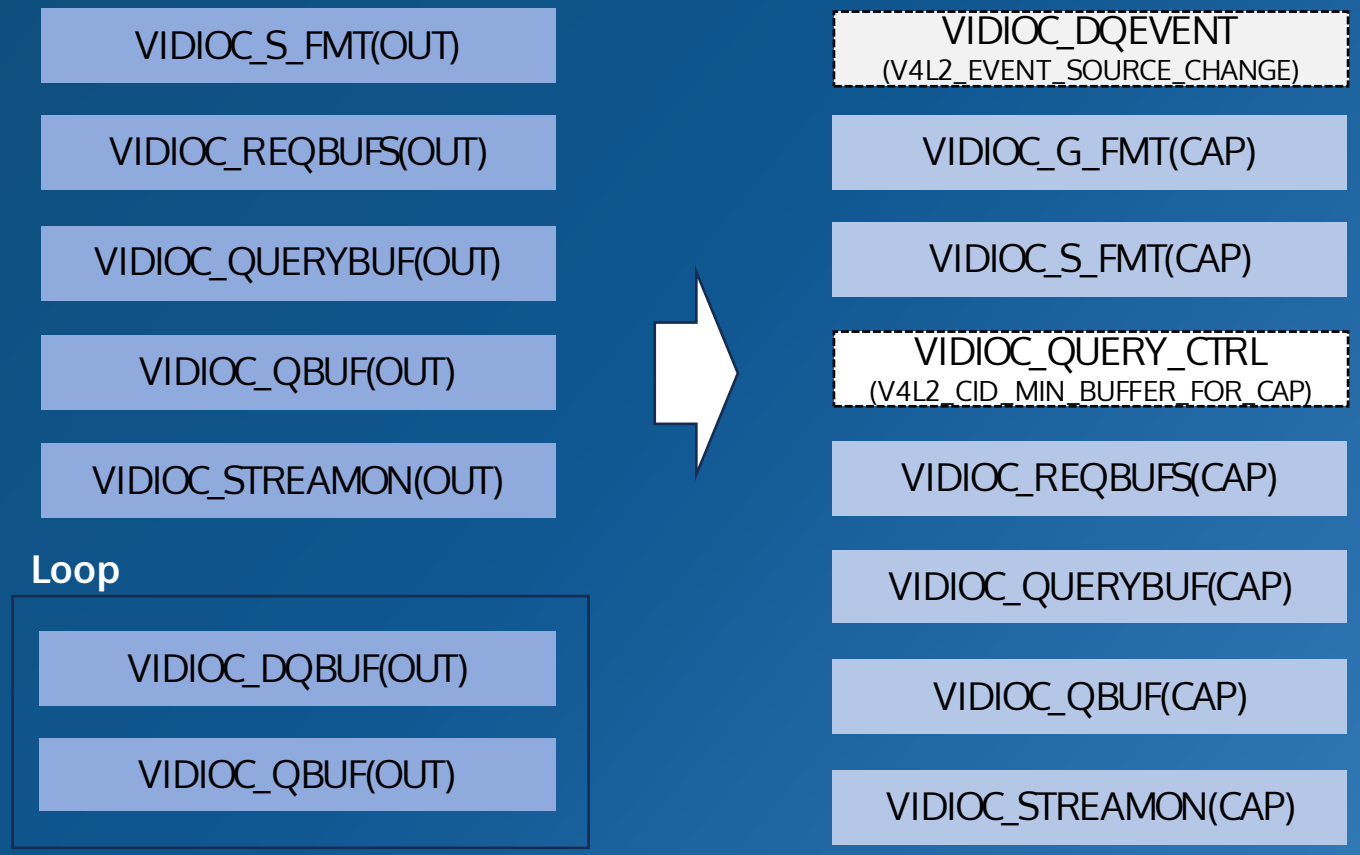
VIDIOC_QUERYBUF(OUT)

VIDIOC_QBUF(OUT)

VIDIOC_STREAMON(OUT)

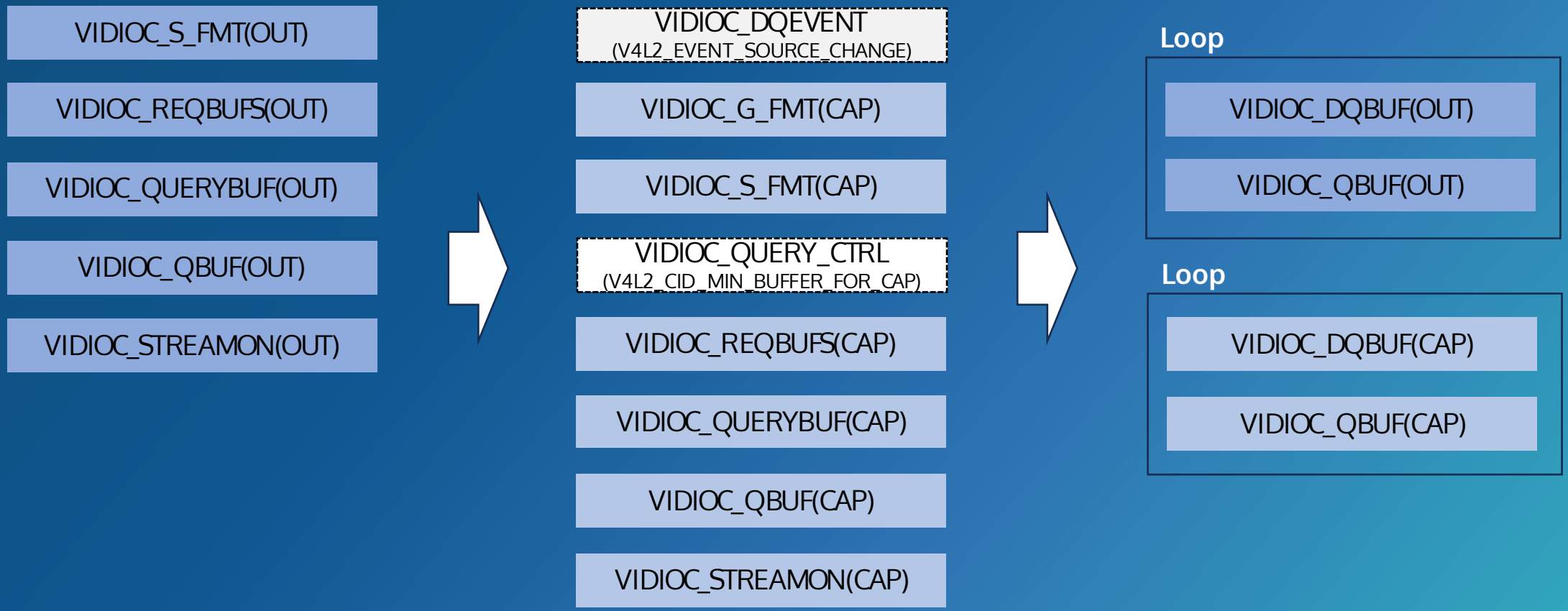
Study V4L2 specification

V4L2 IOCTL flow for stateful decoder



Study V4L2 specification

V4L2 IOCTL flow for stateful decoder



Study V4L2 specification

V4L2 IOCTL flow for stateful encoder

VIDIOC_S_FMT(CAP)

VIDIOC_S_FMT(OUT)

VIDIOC_S_PARM(OUT)

VIDIOC_QUERY_CTRL
(V4L2_CID_MIN_BUFFER_FOR_OUT)

Study V4L2 specification

V4L2 IOCTL flow for stateful encoder

VIDIOC_S_FMT(CAP)

VIDIOC_S_FMT(OUT)

VIDIOC_S_PARM(OUT)

VIDIOC_QUERY_CTRL
(V4L2_CID_MIN_BUFFER_FOR_OUT)

VIDIOC_REQBUFS(OUT)

VIDIOC_ QUERYBUF(OUT)

VIDIOC_STREAMON(OUT)

VIDIOC_ REQBUFS(CAP)

VIDIOC_ QUERYBUF(CAP)

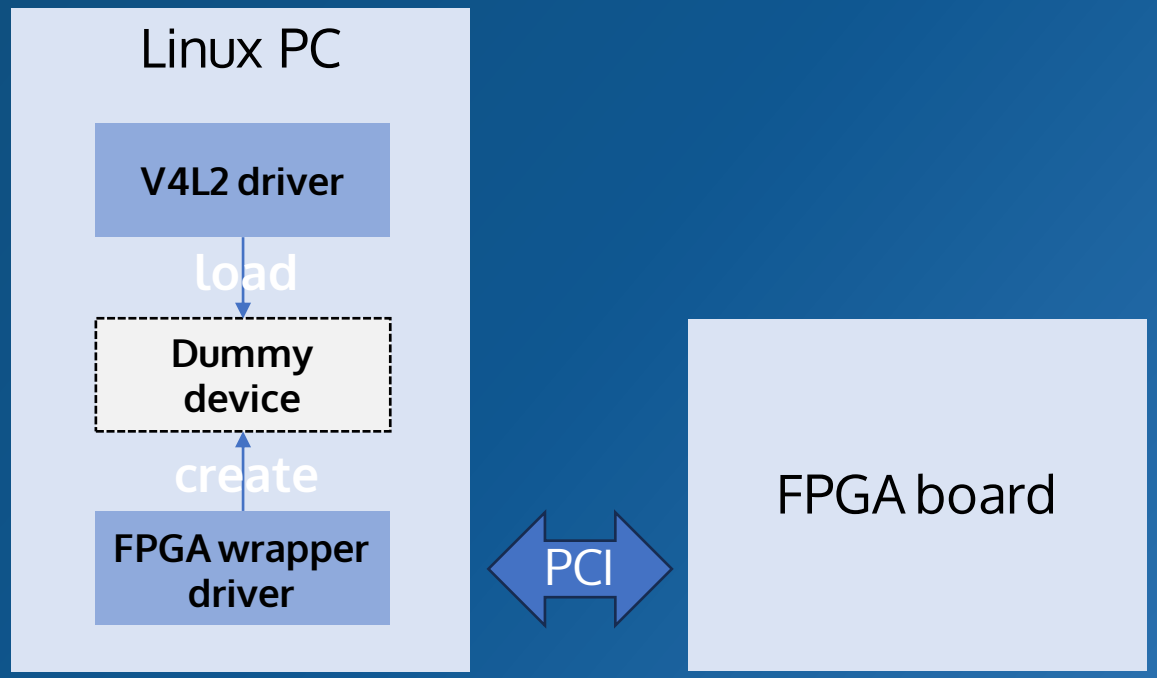
VIDIOC_ STREAMON(CAP)

Study V4L2 specification

V4L2 IOCTL flow for stateful encoder

Initial development on FPGA board

Create dummy device



- PCI driver
- Create dummy device
- Access FPGA board register/memory
- Polling STATUS register

Initial development on FPGA board

Create dummy device

```
fpga_wrapper_probe() {
        struct platform_device_info dummy_info;
        struct platform_device *dummy;
        struct resources[] = {
                { .name = "vpu_irq", .start = dummy_irq, .flags = IORESOURCE_IRQ },
                { .start = pci_resource_start(pdev, 0),
                  .end = pci_resource_start(pdev, 0) + CNM_DEV_REGISTER_SIZE,
                  .flags = IORESOURCE_MEM
                },
        };

        dummy_info.name = “vpu”;
        dummy_info.res = resources;
        dummy_info.num_res = ARRAY_SIZE(resources);
        dummy = platform_device_register_full(&dummy_info);
}
```

Initial development on FPGA board

Create dummy device

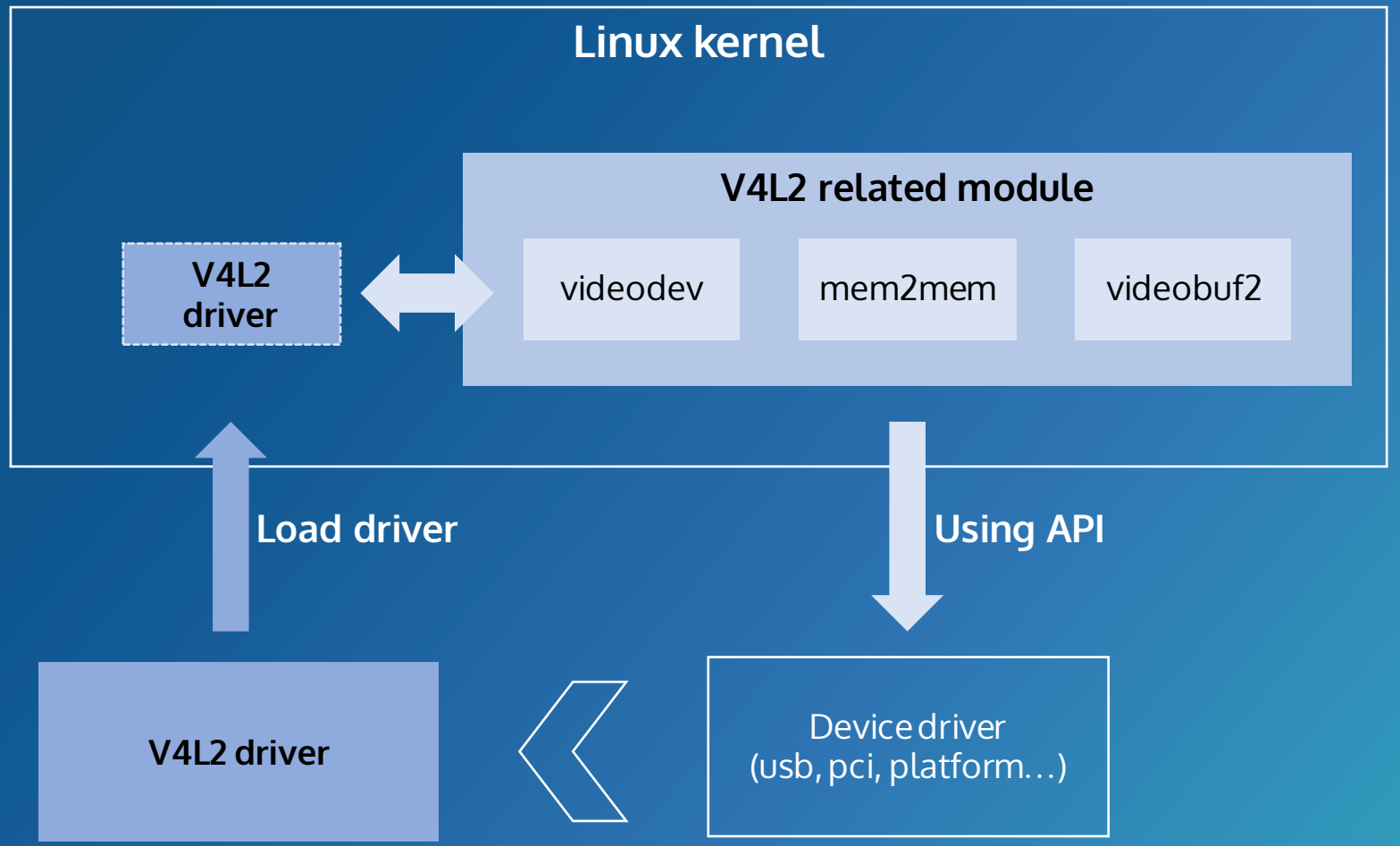
```
fpga_wrapper_probe() {
        int dummy_irq;

        dummy_irq = irq_alloc_descs(-1, 0, 1, 0);

        irq_set_chip_and_handler(dummy_irq, &dummy_irq_chip, handle_simple_irq);
}

polling_status() {
        if (read_register(STATUS))
                generic_handle_irq(dummy_irq);
}
```

Initial development on FPGA board



Initial development on FPGA board

V4L2 core function

```
dummy_device_probe() {
        v4l2_device v4l2_dev;
        video_device *video_dev;

        v4l2_device_register(&v4l2_dev);

        video_dev->fops = &vicodec_fops;
        video_dev->ioctl_ops = &vicodec_ioctl_ops;
        video_dev->v4l2_dev = &v4l2_dev;

        video_register_device(video_dev);
}
```

```
static const struct v4l2_ioctl_ops vicodec_ioctl_ops = {
        .vidioc_querycap        = vidioc_querycap,

        .vidioc_enum_fmt_vid_cap = vidioc_enum_fmt_vid_cap,
        .vidioc_g_fmt_vid_cap   = vidioc_g_fmt_vid_cap,
        .vidioc_try_fmt_vid_cap = vidioc_try_fmt_vid_cap,
        .vidioc_s_fmt_vid_cap   = vidioc_s_fmt_vid_cap,

        .vidioc_g_fmt_vid_cap_mplane    = vidioc_g_fmt_vid_cap,
        .vidioc_try_fmt_vid_cap_mplane  = vidioc_try_fmt_vid_cap,
        .vidioc_s_fmt_vid_cap_mplane    = vidioc_s_fmt_vid_cap,

        .vidioc_enum_fmt_vid_out = vidioc_enum_fmt_vid_out,
        .vidioc_g_fmt_vid_out   = vidioc_g_fmt_vid_out,
        .vidioc_try_fmt_vid_out = vidioc_try_fmt_vid_out,
        .vidioc_s_fmt_vid_out   = vidioc_s_fmt_vid_out,

        .vidioc_g_fmt_vid_out_mplane    = vidioc_g_fmt_vid_out,
        .vidioc_try_fmt_vid_out_mplane  = vidioc_try_fmt_vid_out,
        .vidioc_s_fmt_vid_out_mplane    = vidioc_s_fmt_vid_out,

        .vidioc_reqbufs         = v4l2_m2m_ioctl_reqbufs,
        .vidioc_querybuf        = v4l2_m2m_ioctl_querybuf,
        .vidioc_qbuf            = v4l2_m2m_ioctl_qbuf,
        .vidioc_dqbuf           = v4l2_m2m_ioctl_dqbuf,
        .vidioc_prepare_buf     = v4l2_m2m_ioctl_prepare_buf,
        .vidioc_create_bufs     = v4l2_m2m_ioctl_create_bufs,
        .vidioc_expbuf          = v4l2_m2m_ioctl_expbuf,

        .vidioc_streamon        = v4l2_m2m_ioctl_streamon,
        .vidioc_streamoff       = v4l2_m2m_ioctl_streamoff,
```

Initial development on FPGA board

V4L2 core function

- m2m ioctl helper function
  - Defined v4l2_m2m_ioctl_XXX
  - Call vb2 core helper function internally

- Call back function for v4l2_m2m_ops
  - device_run() : Begin actual job
  - job_ready()
    - One source buffer and one destination buffer are ready
    - Called before device_run()
  - job_abort() : Abort running job

Initial development on FPGA board

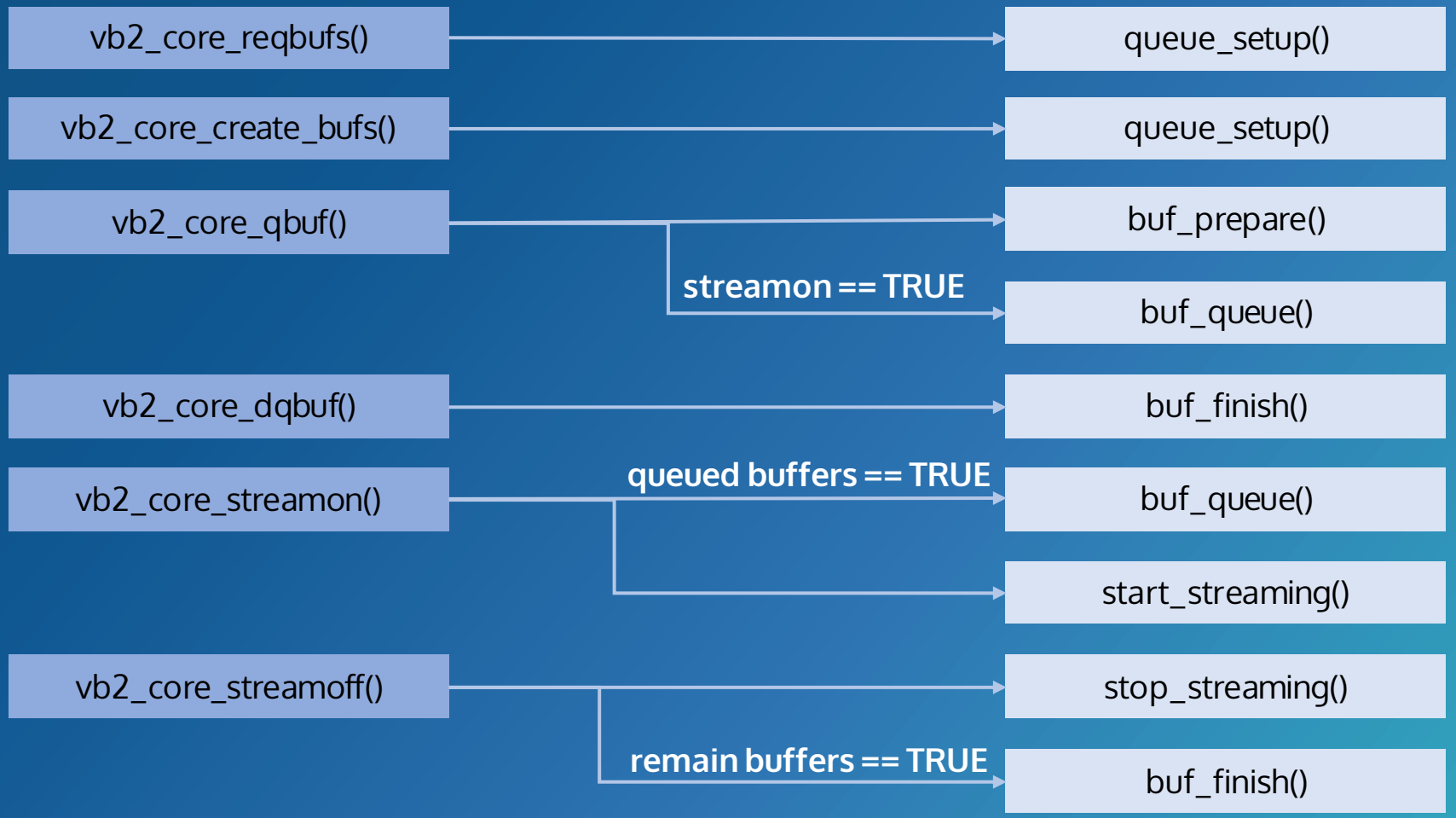
V4L2 core function

- vb2 ioctl helper function
  - Defined vb2_ioctl_xxx
  - Call vb2 core helper function internally

- Call back function for vb2_ops
  - queue_setup()
    - Called before memory allocation
    - Check required number of buffer in driver
  - start_streaming() : Called once to enter ‘streaming’ state
  - stop_streaming() : Called when ‘streaming’ state must be disabled
  - buf_queue() : Passes buffer to the driver

Initial development on FPGA board

V4L2 core function



Initial development on FPGA board

V4L2 core function

- Debugging
  - echo 3 > /sys/class/video4linux/video0/dev_debug
  - echo Y > /sys/module/v4l2_mem2mem/parameters/debug
  - echo 3 > /sys/module/videobuf2_v4l2/parameters/debug
  - echo 3 > /sys/module/videobuf2_common/parameters/debug

Initial development on FPGA board

V4L2 test tools

- v4l2-ctl
  - An application to control v4l2 drivers
  - Managed by linux-media maintainers
  - Test video decoder/encoder in command line
    - --device=/dev/videoX
    - --set-fmt-video-out=pixelformat=HEVC (for decoder)
    - --set-fmt-video=pixelformat=HEVC (for encoder)
    - --stream-mmap
    - --stream-out-mmap
    - --stream-from=src.bin
    - --stream-to=dst.yuv
    - --stream-count=10

Initial development on FPGA board

V4L2 test tools

- v4l2-compliance
  - An application to test v4l2 drivers
  - Managed by linux-media maintainers
  - Test almost all V4L2 ioctls

```
Required ioctls:
        test VIDIOC_QUERYCAP: OK
        test invalid ioctls: OK

Allow for multiple opens:
        test second /dev/video1 open: OK
        test VIDIOC_QUERYCAP: OK
        test VIDIOC_G/S_PRIORITY: OK
        test for unlimited opens: OK

Debug ioctls:
        test VIDIOC_DBG_G/S_REGISTER: OK (Not Supported)
        test VIDIOC_LOG_STATUS: OK (Not Supported)

Input ioctls:
        test VIDIOC_G/S_TUNER/ENUM_FREQ_BANDS: OK (Not Supported)
        test VIDIOC_G/S_FREQUENCY: OK (Not Supported)
        test VIDIOC_S_HW_FREQ_SEEK: OK (Not Supported)
        test VIDIOC_ENUMAUDIO: OK (Not Supported)
        test VIDIOC_G/S/ENUMINPUT: OK (Not Supported)
        test VIDIOC_G/S_AUDIO: OK (Not Supported)
        Inputs: 0 Audio Inputs: 0 Tuners: 0

Output ioctls:
        test VIDIOC_G/S_MODULATOR: OK (Not Supported)
        test VIDIOC_G/S_FREQUENCY: OK (Not Supported)
        test VIDIOC_ENUMAUDOUT: OK (Not Supported)
        test VIDIOC_G/S/ENUMOUTPUT: OK (Not Supported)
        test VIDIOC_G/S_AUDOUT: OK (Not Supported)
        Outputs: 0 Audio Outputs: 0 Modulators: 0

Input/Output configuration ioctls:
        test VIDIOC_ENUM/G/S/QUERY_STD: OK (Not Supported)
        test VIDIOC_ENUM/G/S/QUERY_DV_TIMINGS: OK (Not Supported)
        test VIDIOC_DV_TIMINGS_CAP: OK (Not Supported)
        test VIDIOC_G/S_EDID: OK (Not Supported)

Control ioctls:
        test VIDIOC_QUERY_EXT_CTRL/QUERYMENU: OK
        test VIDIOC_QUERYCTRL: OK
        test VIDIOC_G/S_CTRL: OK
```

Verification on TI board

Load driver

- Linux kernel build
  - Cross compiler
  - Enable Multimedia support config option
- Device Tree
  - A data structure that describes the hardware
  - In linux/arch/arm64/boot/dts/ti/k3-j721s2-main.dtsi

```
vpu: video-codec@4210000 {
        compatible = "ti,j721s2-wave521c", "cnm,wave521c";
        reg = <0x00 0x4210000 0x00 0x10000>;
        interrupts = <GIC_SPI 182 IRQ_TYPE_LEVEL_HIGH>;
        clocks = <&k3_clks 179 2>;
        power-domains = <&k3_pds 179 TI_SCI_PD_EXCLUSIVE>;
};
```

Verification on TI board

Load driver

- Device Tree documentation format
  - In Documentation/devicetree/bindings/media/
  - YAML format

```
# SPDX-License-Identifier: (GPL-2.0 OR BSD-2-Clause)
%YAML 1.2
---
$id: http://devicetree.org/schemas/media/cnm,wave521c.yaml#
$schema: http://devicetree.org/meta-schemas/core.yaml#

title: Chips&Media Wave 5 Series multi-standard codec IP

maintainers:
  - Nas Chung <nas.chung@chipsnmedia.com>
  - Jackson Lee <jackson.lee@chipsnmedia.com>

description:
  The Chips&Media WAVE codec IP is a multi format video encoder/decoder

properties:
  compatible:
    items:
      - enum:
          - ti,j721s2-wave521c
      - const: cnm,wave521c

  reg:
    maxItems: 1

  clocks:
    items:
      - description: VCODEC clock

  interrupts:
    maxItems: 1

  power-domains:
    maxItems: 1
```

Verification on TI board

Load driver

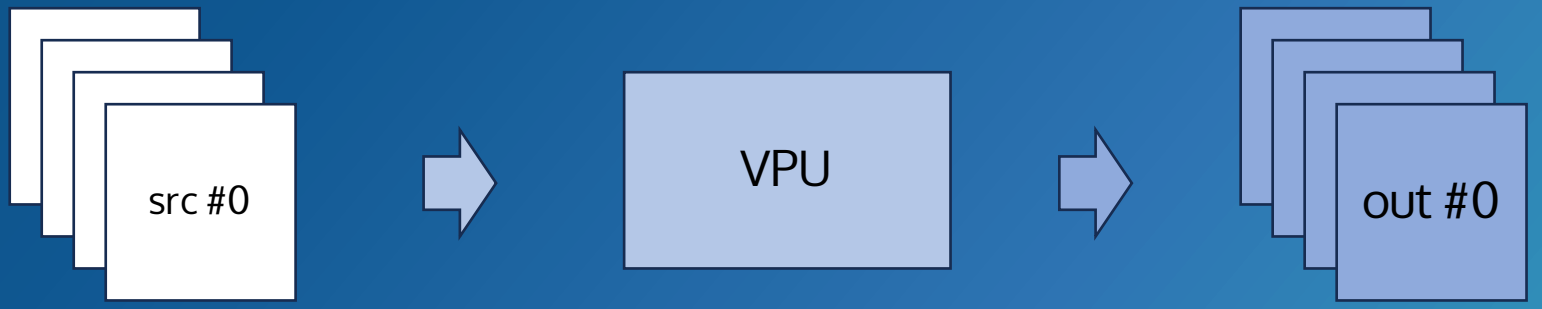
```
static const struct of_device_id wave5_dt_ids[] = {
        { .compatible = "ti,j721s2-wave521c", .data = &ti_wave521c_data },
        { /* sentinel */ }
};
MODULE_DEVICE_TABLE(of, wave5_dt_ids);

static struct platform_driver wave5_vpu_driver = {
        .driver = {
                .name = VPU_PLATFORM_DEVICE_NAME,
                .of_match_table = of_match_ptr(wave5_dt_ids),
                },
        .probe = wave5_vpu_probe,
        .remove_new = wave5_vpu_remove,
};
```

Verification on TI board

Interrupt handling

- C&M command queue
  - Host send multiple command
  - Do not wait previous command done



Verification on TI board

Interrupt handling

- IRQ handle
  - Read interrupt reason
  - Queue interrupt reason to kfifo
  - Clear interrupt reason

- Threaded IRQ handler
  - Dequeue interrupt reason for kfifo
  - Handle decoding/encoding result
  - Handle mutex

Verification on TI board

Interrupt handling

```
probe() {
        ret = devm_request_threaded_irq(&pdev->dev, dev->irq, vpu_irq,
                                        vpu_irq_thread, 0, "vpu_irq", dev);
}

static irqreturn_t vpu_irq(int irq, void *dev_id) {
        if (read_reg(STATUS)) {
                reason = readl(REASON);
                kfifo_in(&dev->reason_q, &reason, sizeof(int));
                return IRQ_WAKE_THREAD;
        }
        return IRQ_HANDLED;
}
```

Verification on TI board

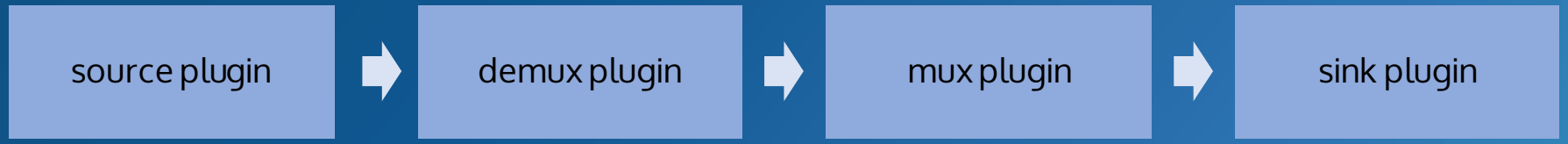
Interrupt handling

```
static irqreturn_t vpu_irq_thread(int irq, void *dev_id) {
        while (kfifo_len(&dev->reason_q)) {
                kfifo_out(&dev->reason_q, &reason, sizeof(int));
                ops->finish_process(inst);
        }
        return IRQ_HANDLED;
}
```

Verification on TI board

Gstreamer

- Open source multimedia framework
- Pipeline



- Support V4L2 plugin
  - In gst-plugins-good

```
$ gst-inspect-1.0 | grep v4l2
video4linux2: v4l2h265enc: V4L2 H.265 Encoder
video4linux2: v4l2h264enc: V4L2 H.264 Encoder
video4linux2: v4l2h265dec: V4L2 H265 Decoder
video4linux2: v4l2h264dec: V4L2 H264 Decoder
video4linux2: v4l2deviceprovider (GstDeviceProviderFactory)
video4linux2: v4l2radio: Radio (video4linux2) Tuner
video4linux2: v4l2sink: Video (video4linux2) Sink
video4linux2: v4l2src: Video (video4linux2) Source
```

Verification on TI board

Gstreamer

- gst-launch-1.0

```
// For decoder
$ gst-launch-1.0 filesrc location=./akiyo.cfg_ramain_tv0.cfg.265 ! h265parse ! v4l2h265dec ! filesink location=./gst_test.yuv
$ gst-launch-1.0 filesrc location=./bd_8b_01.cfg_0.264 ! h264parse ! v4l2h264dec ! filesink location=./gst_test.yuv

// For encoder
$ gst-launch-1.0 filesrc location=./BasketballPass_416x240_50.yuv ! rawvideoparse width=416 height=240 format=i420 framerate=30/1 ! v4l2h265enc ! filesink location=./gst_test.bin
$ gst-launch-1.0 filesrc location=./BasketballPass_416x240_50.yuv ! rawvideoparse width=416 height=240 format=i420 framerate=30/1 ! v4l2h264enc ! filesink location=./gst_test.bin

// For encoder from gstreamer version 1.20.5
$ gst-launch-1.0 filesrc location=./foreman_cif.yuv ! rawvideoparse width=352 height=288 format=i420 framerate=30/1 colorimetry=bt601 ! v4l2h264enc ! filesink location=./gst_test_h264.bin
```

- gst-validate-1.0
  - Testing seek, stop, resume, dynamic parameter change…
  - Refer scenario file
    - seek=true, duration=5.0, min-media-duration=4.0
    - seek, playback-time=1.0, rate=1.0, start=2.0, flags=accurate+flush

Verification on TI board

Gstreamer

- Debugging
  - export GST_DEBUG=X (log level)
    - 5 (DEBUG) : Logs all debug messages
    - 6 (LOG) : Logs all log messages
  - export GST_DEBUG_FILE=XXX.txt
  - GST_DEBUG=[object_name]:[debug_level] gst-launch-1.0 ~
    - Show log for specific object

Verification on TI board

Fluster test

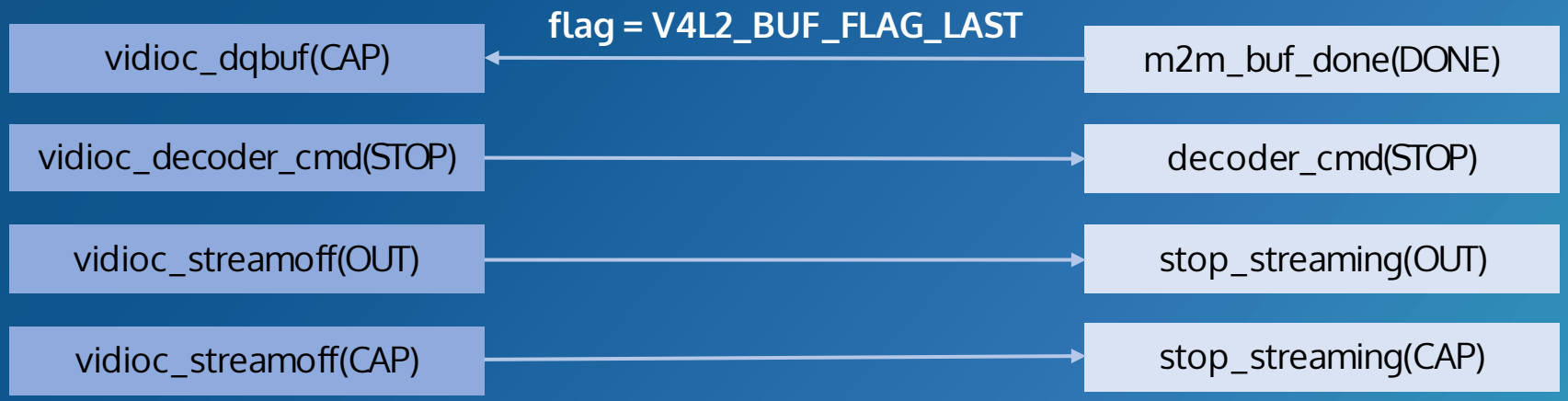
- A testing framework for decoder conformance
  - H.265/HEVC, H.264/AVC, H.266/VVC, VP8, VP9, AV1 and AAC
  - Testing ISO/JCT/JVT... Streams
  - Compare output (YUV) data

- Command
  - python3 ./fluster.py -d GStreamer-H.264-V4L2-Gst1.0 -j1 -ts JVT-AVC_V1
    - Testing all JVT-AVC_V1 test suite
  - python3 ./fluster.py -ts JCT-VC-HEVC_V1 -tv BUMPING_A_ericsson_1
    - Testing only BUMPING_A_ericsson_1 stream

Verification on TI board

V4L2 finish sequence

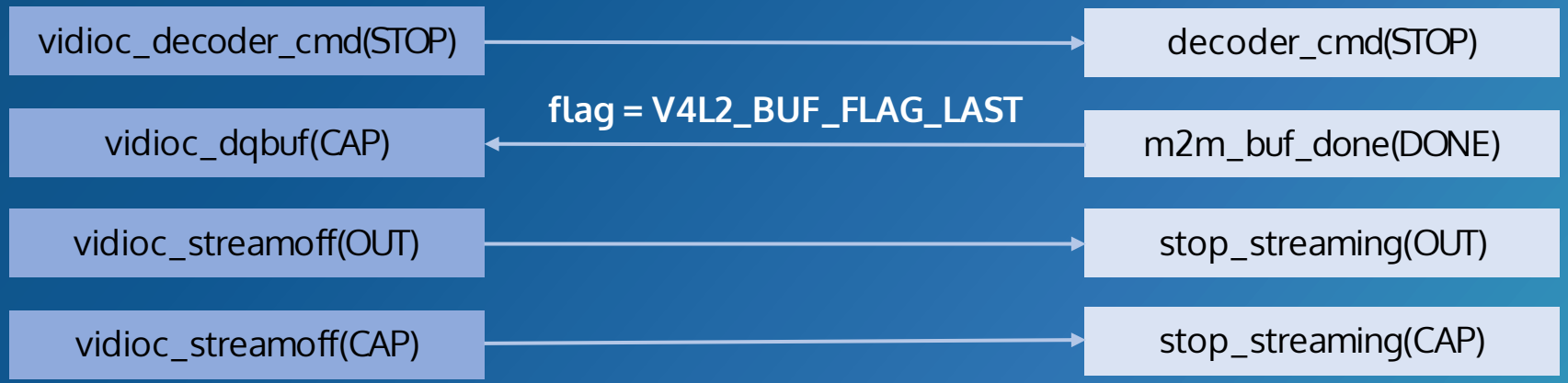
- Normal case



Verification on TI board

V4L2 finish sequence

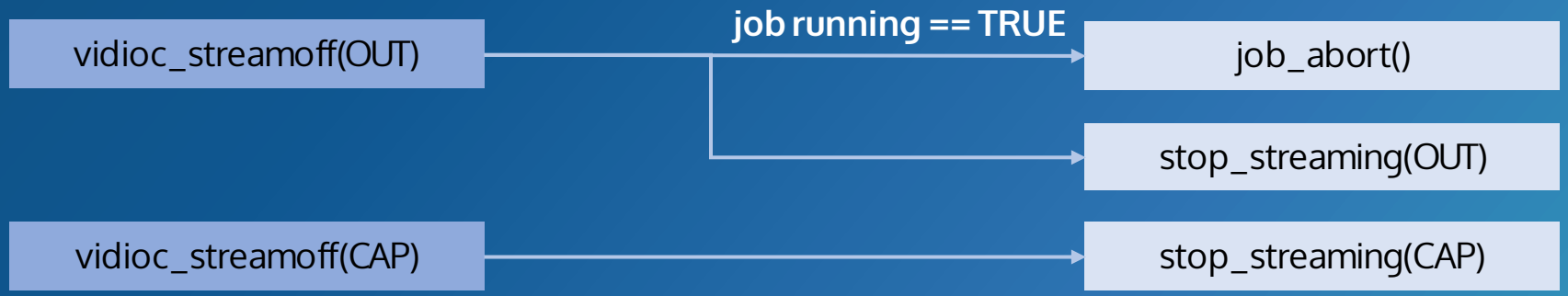
- Normal case



Verification on TI board

V4L2 finish sequence

- Abnormal case



Verification on TI board

Error handling

- Recoverable error
  - Set V4L2_BUF_FLAG_ERROR flag
  - Decoding/Encoding may continue as normal
- Fatal failure
  - Does not allow the decoding/encoding to continue
  - Report -EIO error for all IOCTL function

```
vb2_queue_error(v4l2_m2m_get_src_vq(m2m_ctx));
vb2_queue_error(v4l2_m2m_get_dst_vq(m2m_ctx));
```

Understanding upstream process

Linux kernel contribution

- Linux kernel is open source
  - Development cycle
  - Merge window
    - Stable code is merged into the mainline
    - Approximately 2 weeks
- First step
  - Read https://docs.kernel.org/process/~
  - Subscribe mailing list
  - Get familiar with Git

Understanding upstream process

Linux kernel contribution

1. Get the latest linux-kernel source tree
2. Change source codes
   - check coding style
3. Commit your changes
   - follow canonical patch format
4. Create patch files
   - follow canonical patch format
5. Send mail to maintainers
6. Communicate with reviewers

Understanding upstream process

Check coding style

- Linux kernel coding style
  - https://www.kernel.org/doc/html/latest/process/coding-style.html
  - Tabs/Indentations are 8 characters
  - Always uses braces for multiple statement
  - The preferred limit on the length of a single line is 80 columns
  - …
- Run checkpatch.pl
  - Checks for trivial style violations in patches or source files
  - In linux/scripts/checkpatch.pl
    - checkpatch.pl -f *.c *.h

Understanding upstream process

Check coding style

- Avoid unnamed numerical constants
- Remove dead code
- Remove duplicated code
  - Utilize V4L2 core helper function
- Add comments
  - Tell WHAT your code does, not HOW

Understanding upstream process

Follow canonical patch format

```
commit 60fb58f7312735a85a5fbd9983d40f2577880b41
Author: Nas Chung <nas.chung@chipsnmedia.com>
Date:   Mon Apr 25 18:25:56 2022 +0900

    staging: media: wave5: Add vpuapi layer

    Add the vpuapi layer of the wave5 codec driver.
    This layer is used to configure the hardware according
    to the parameters.

    Signed-off-by: Dafna Hirschfeld <dafna.hirschfeld@collabora.com>
    Signed-off-by: Robert Beckett <bob.beckett@collabora.com>
    Signed-off-by: Nas Chung <nas.chung@chipsnmedia.com>
```

- Subject : subsystem: summary phrase
- Body
  - From: Author name <Author email>
  - Describe this commit
  - An empty line
  - Signed-off-by: name <email>
    - To improve tracking of who did what
    - Certifies the right to open-source patch

Understanding upstream process

Follow canonical patch format

```
$ git format-patch --cover-letter -v 7 HEAD~6
v7-0000-cover-letter.patch
v7-0001-staging-media-wave5-Add-vpuapi-layer.patch
v7-0002-staging-media-wave5-Add-the-vdi-layer.patch
v7-0003-staging-media-wave5-Add-the-v4l2-layer.patch
v7-0004-staging-media-wave5-Add-TODO-file.patch
v7-0005-dt-bindings-media-staging-wave5-add-yaml-devicetr.patch
v7-0006-media-wave5-Add-wave5-driver-to-maintainers-file.patch
```

- Cover letter
  - Describe multiple patch series
  - Include test report
  - Include version history
- Version
  - Version of this patch series
- Run checkpatch.pl

Understanding upstream process

Send mail

```
$ git send-email --envelope-sender=nas.chung@chipnsmedia.com --to linux-media@vger.kernel.org --cc hverkuil@xs4all.nl,nas.chung@chipsnmedia.com --annotate *.patch
```

- Install
  - git-email / libmailtools-perl
- Mail format : Just plain text
- Check maintainers
  - In linux/scripts/get_maintainer.pl
    - get_maintainer.pl drivers/media/platform/chips-media/wave5
- Option
  - --envelope-sender : specify the envelope sender used to send the emails
  - --annotate : review and edit mail before sending

Understanding upstream process

Communicate with reviewers

- Check patchwork for subsystem project
- Receive comments(feedback) within a few weeks
  - Be patient
- Should respond to reviewers
  - Thank them for their time
  - Interleaved reply style

Understanding upstream process

Communicate with reviewers

- ACK : I agree
- NACK : I disagree
- LGTM : Looks good to me
- IMO : In my opinion
- IMHO : In my humble/honest opinion
- Nit : Nitpick, trivial issues
- BTW : By the way

THANK YOU

marketing@chipsnmedia.com
Tel) +82.2.568.3767

Visit www.chipsnmedia.com for more information